

Examen XML

— Master d'Ingénierie Informatique —

Mars 2010, durée 2h.

Les réponses apportées aux questions doivent être justifiées avec clarté et concision. Les documents sont interdits à l'exception d'une feuille de memento (4 pages A4). L'examen est composé

```
<xsd:complexType name="TypeA" abstract="true">
```

```
...
```

```
</xsd:complexType>
```

Est-il encore possible d'utiliser l'élément `elemA` dans un document valide ? Si oui, expliquer comment.

- e) Donner un document valide pour le schéma obtenu en remplaçant les déclarations des éléments `elemA` et `elemB` par les déclarations suivantes.

► Exercice 4 On considère le document suivant inspiré des données Gedcom.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<gedcom>
<indi id="I1">
  <name>Victoria /Hanover/</name>
  <titl>Queen of England</titl>
  <fams idref="F1"/>
  <famc idref="F42"/>
</indi>
<indi id="I70">
  <name>Bessiewallis /Warfield/</name>
  <fams idref="F20"/>
  <fams idref="F24"/>
  <fams idref="F25"/>
  <famc idref="F55"/>
</indi>
<indi id="I87">
  <name>William Henry Andrew /Windsor/</name>
  <titl>Prince</titl>
  <sex>M</sex>
  <famc idref="F19"/>
</indi>
</gedcom>
```

a) Donner le résultat de l'application de la feuille de style XSLT suivante sur le document précédent.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" encoding="iso-8859-1" indent="yes"/>
  <xsl:template match="/">
    <xsl:copy>
      <xsl:copy-of select="@*" />
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>
  <xsl:template match="indi">
    <xsl:copy>
      <xsl:copy-of select="@*" />
      <xsl:apply-templates select="*[name() != 'fams']" />
      <xsl:if test="count(fams) = 1">
        <fams idref="{fams/@idref}" />
      </xsl:if>
      <xsl:if test="count(fams) > 1">
        <fams>
          <xsl:attribute name="idrefs">
            <xsl:value-of select="fams/@idref"/>
          </xsl:attribute>
        </fams>
      </xsl:if>
    </xsl:copy>
  </xsl:template>
</xsl:stylesheet>
```

b) Donner une feuille de style qui effectue la transformation inverse. On pourra utiliser la fonction XPath tokenize qui découpe la chaîne passée en premier paramètre à chaque

occurrence de l'expression rationnelle passée en second paramètre. Utiliser par exemple '\s+' comme second paramètre.